改进布谷鸟算法求解双资源约束柔性车间调度问题 *

罗浩嘉 [a]，潘大志 [a,b†]
(西华师范大学 a. 数学与信息学院; b. 计算方法与应用研究所，四川 南充 637009)

摘 要：针对双资源约束的柔性车间调度问题(DRCFJSP)，以优化最大完工时间为目标，设计出一种具有改进解码方案的布谷鸟算法对其进行求解。由于 DRCFJSP 除了需要考虑机器的分配，还需要兼顾工人的加工情况，因此改进了传统解码方式以避免机器和工人在加工时间上的冲突，同时在解码时尽可能利用机器和工人的空闲时间。在布谷鸟算法核心框架下，将布谷鸟种群随机划分为 3 个子群，每个子群采用不同 Lévy 飞行方式独立进行寻优，并通过差分算子实现子群间信息交流，不仅增强了算法的全局搜索能力也平衡了算法的局部搜索能力。最后通过基准测试算例进行实验仿真分析并与其他算法进行对比，验证了改进布谷鸟算法和改进解码方法的有效性优越性。

关键词：柔性车间调度；双资源约束；布谷鸟算法；改进解码方法
中图分类号：TP301.6　　doi: 10.19734/j.issn.1001-3695.2022.01.0030

Improved cuckoo algorithm for flexible job shop scheduling with dual resource constraints

Luo Haojia[a], Pan Dazhi[a, b†]
(*a. School of Mathematics & Information*, *b. Institute of Computing Methods & Applications China West Normal University*, *China West Normal University*, *Nanchong Sichuan* 637009, *China*)

Abstract: Aiming at the flexible job shop scheduling problem with dual resource constraints (DRCFJSP) , this paper designs a cuckoo algorithm with an improved decoding scheme to solve it with the goal of optimizing the maximum completion time. Since DRCFJSP needs to consider the allocation of machines and the processing situation of workers, this paper improves the traditional decoding method to avoid the conflict of processing time between machines and workers, and makes use of the idle time of machines and workers as much as possible during decoding. Under the core framework of the cuckoo algorithm, this paper divides the cuckoo population into three subpopulations randomly, and each subpopulation adopts different Lévy flight methods to search for the optimum independently, and realizes the information exchange between subpopulations through the difference operator, which not only enhances the global search ability of the algorithm but also balances the local search ability of the algorithm. Through the experimental simulation analysis of the benchmark test case, the results show the effectiveness and superiority of the improved cuckoo algorithm and the improved decoding method.

Key words: flexible job shop scheduling; dual resource constraints; cuckoo algorithm; improved decoding method

0 引言

调度在制造业和服务行业中起着至关重要的作用，作业车间调度问题作为调度中的典型问题，在制造系统领域得到了广泛的关注。近些年也有很多基于作业车间调度的扩展研究，比如带模糊加工时间的车间调度[1,2]、带阻塞问题的车间调度[3~5]和多目标车间调度[6,7]等，这些扩展研究都是车间调度问题在实际情况中的应用。双资源约束柔性作业车间调度问题(Dual Resource Constrained Flexible Job-shop Scheduling Problem，DRCFJSP)是柔性作业车间调度问题(Flexible Job-Shop Scheduling Problem，FJSP)的一个扩展，在 FJSP 的基础上增加了工人这一约束条件,在进行加工时需要兼顾工人和机器的加工情况，因此 DRCFJSP 比 FJSP 更接近实际生产情况。近些年来，也有一些元启发式算法对 DRCFJSP 进行研究，如 Li 等[8]提出了一种分支种群遗传算法，引入了基于扇区分割的轮盘选择算子，利用精英进化算子的优化搜索能力，有效降低了算法复杂度，避免了算法早熟。Zhang 等[9]针对具有资源灵活性的双资源约束作业车间调度问题，提出了一种新颖的混合离散粒子群优化算法，引入了具有可变邻域结构的改进模拟退火算法，提高了算法的局部搜索能力。Lei 等[10]提出了一种有效的变量邻域搜索，通过两个邻域搜索程序依次执行，分别为两个子问题产生新的解决方案，从而增强算法的搜索能力。Zheng 等[11]提出了一种具有新编码方案的知识引导果蝇优化算法来求解最小化完工时间的 DRCFJSP，将知识引导搜索和基于气味搜索相结合，平衡了算法全局探索和局部开发能力。

布谷鸟搜索算法(cuckoo search，CS)[12]是由剑桥大学学者 Yang 和 Deb 于 2009 年通过模拟布谷鸟寄生育雏行为提出的一种新型元启发式搜索算法。由于其所用参数少、全局搜索能力强等优点，被普遍应用于连续性优化问题、调度问题、工程优化问题等多个领域。Ouaar 等[13]在不改变布谷鸟算法框架的情况下，将布谷鸟算法离散化，用于求解车间调度问题。ALAA 等[14]改进 CS 算法中的 Lévy 飞行，加强对种群最优个体领域的搜索，并将改进后的算法用于求解柔性车间调度问题。Majumdera 等[15]针对作业准备时间不相等的并行批处理机调度问题，提出了一种混合离散布谷鸟搜索(HDCS)算法来求解其最小完工时间。他们通过将 CS 算法与变量邻域搜索算法结合，构造了该混合算法，并且对 Lévy 飞行方式进行了改进，对算法的搜索进行了优化。唐红涛等[16]在考虑到企业的实际生产情况下，建立了一个调度目标为最大完工时

收稿日期：2022-01-29；**修回日期：**2022-03-17　**基金项目：**国家自然科学基金资助项目(11871059)；四川省教育厅自然科学基金资助项目(18ZA0469)；西华师范大学英才科研基金资助项目(17YC385)

作者简介：罗浩嘉(1997-)，女，四川遂宁人，硕士研究生，主要研究方向为智能算法、数学建模；潘大志(1974-)，男(通信作者)，四川三台人，教授，硕导，博士，主要研究方向为智能计算，算法设计等(pdzzj@126.com).

间最小的分布式柔性车间调度模型，通过对布谷鸟算法的编码和初始化策略进行调整以提高初始解的质量，同时对布谷鸟算法的搜索操作进行了改进，在加快算法的收敛速度的同时保证解的质量。罗函明等[17]在求解混合流水车间调度问题时，将布谷鸟算法进行离散化，并改进其解码方式，提高解的质量。

目前，有关 DRCFJSP 问题的研究并不算多，但大部分工厂在进行生产时，工人的配合却是必不可少的，因此本文在 FJSP 问题的基础上，考虑到工厂的实际加工情况，增加了工人约束，通过工人操控机器对工件进行加工。针对以上问题，本文在编码时增加了工人信息，同时将种群分为 3 个子群，对子群的步长因子进行自适应调整，提高种群多样性。最后对算法解码进行了改进，将工人信息考虑进了算法解码中，并且充分利用解码时产生的空闲时间，以达到缩短算法最终完工时间的目的。通过标准测试集对改进后的算法进行了仿真实验，并与其他算法进行了对比，证明了改进 CS 算法求解 DRCFJSP 问题的有效性与稳定性。

1　双资源约束柔性车间调度模型

双资源约束的柔性车间调度问题(DRCFJSP)描述如下：w 个工人 $W=\{W_1,W_2,\ldots,W_w\}$ 操作 m 台机器 $M=\{M_1,M_2,\ldots,M_m\}$ 对 n 个工件 $J=\{J_1,J_2,\ldots,J_n\}$ 进行加工，工件 $J_i(i=1,2,\ldots,n)$ 包含确定的 n_i 道工序 $\{O_{i,1},O_{i,2},\ldots,O_{i,n_i}\}$。每个操作的加工时间取决于机器和工人分配，不同的机器和工人可能会导致加工时间不同。调度的目标是确定所有工件的加工顺序以及每道工序选用的工人和机器，使得最大完工时间(makespan)最小。DRCFJSP 中的符号定义如表 1 所示。

表 1　符号变量定义

Tab. 1　Symbol variable definition

符号	定义
n	工件总数
m	机器总数
w	工人总数
M	总的机器集
W	总的工人集
$O_{i,j}$	工件 i 的第 j 道工序
$SE_{i,j}$	工序 $O_{i,j}$ 最早开始加工时间
$CE_{i,j}$	工序 $O_{i,j}$ 最早完工时间
$SL_{i,j}$	工序 $O_{i,j}$ 最晚开始加工时间
$CL_{i,j}$	工序 $O_{i,j}$ 最晚完工时间
S_k	机器 k 的开始加工时间
F_k	机器 k 的结束加工时间
L	一个足够大的正数
T_{ijkw}	工序 $O_{i,j}$ 在机器 k 上由工人 w 加工所需时间
$PW[v]$	工序 v 由同一工人加工的前一道工序，若 v 为该工人加工的第一道工序，则 $PW[v]=$-1
$SW[v]$	工序 v 由同一工人加工的后一道工序，若 v 为该工人加工的最后一道工序，则 $SW[v]=$-1

决策变量如下:

$$x_{ijkw}=\begin{cases}1, & \text{如果工序}O_{ij}\text{ 在机器}k\text{ 上由工人}w\text{ 加工}\\ 0, & \text{否则}\end{cases}$$

$$y_{ghijk}=\begin{cases}1, & \text{如果工序}O_{g,h}\text{ 与工序}O_{i,j}\text{ 都在机器}k\text{ 上加工，且工序 }O_{g,h}\text{ 先于工序}O_{i,j}\\ 0, & \text{否则}\end{cases}$$

$$z_{ghijw}=\begin{cases}1, & \text{如果工序}O_{g,h}\text{ 与工序}O_{i,j}\text{ 都由工人}w\text{ 加工，且工序 }O_{g,h}\text{ 先于工序}O_{i,j}\\ 0, & \text{否则}\end{cases}$$

以最小化最大完工时间为优化目标建立数学模型，其目标函数为

$$\min\ C_{\max}=\max\left(CE_{i,j}\right)$$

其约束条件如下：

$$\forall i\in J,\forall j\in J_i,\sum_{k\in M}\sum_{w\in W}x_{ijkw}=1 \tag{1}$$

$$\forall i\in J,\forall j\in J_i,CE_{i,j}=SE_{i,j}+\sum_{k\in M}\sum_{w\in W}\left(T_{ijkw}x_{ijkw}\right) \tag{2}$$

$$\forall i\in J,\forall j\in J_i,CL_{i,j}=SL_{i,j}+\sum_{k\in M}\sum_{w\in W}\left(T_{ijkw}x_{ijkw}\right) \tag{3}$$

$$\forall i\in J,\forall j\in J\{1,2,\ldots,n_i-1\},CE_{i,j}\le SE_{i,j+1} \tag{4}$$

$$\forall g,i\in J,\forall h\in J_g,\forall j\in J_i,\forall k\in M,\ CE_{g,h}\le SE_{i,j}+L\left(1-y_{ghijk}\right) \tag{5}$$

$$\forall g,i\in J,\forall h\in J_g,\forall j\in J_i,\forall w\in W,CE_{g,h}\le SE_{i,j}+L\left(1-z_{ghijw}\right) \tag{6}$$

$$\forall k\in M,S_k\ge 0 \tag{7}$$

$$\forall k\in M,\forall i\in J,\forall j\in J_i,\forall w\in W,S_k\le SE_{i,j}+L\left(1-x_{ijkw}\right) \tag{8}$$

$$\forall k\in M,\forall i\in J,\forall j\in J_i,\forall w\in W,CE_{i,j}\cdot x_{ijkw}\le F_k \tag{9}$$

若 $PW[O_{i,j}]=-1$，令 $CE_{PW[O_{i,j}]}=0$，则有 $\forall i\in J,\forall j\in J_i$，

$$SE_{i,j}=\max(CE_{PM[O_{i,j}]},CE_{PJ[O_{i,j}]},\sum_{k\in M}\sum_{w\in W}S_k\cdot x_{ijkw}) \tag{10}$$

若 $SW[O_{i,j}]=-1$，令 $SL_{SW[O_{i,j}]}=C_{\max}$，则有 $\forall i\in J,\forall j\in J_i$，

$$CL_{i,j}=\max(SL_{SM[O_{i,j}]},SL_{SJ[O_{i,j}]},SL_{SW[O_{i,j}]}) \tag{11}$$

式(1)表示每个工序只能在一台机器上由一个工人进行加工；式(2)表示每个工序的最早完工时间等于该工序最早开始加工时间与所需加工时间之和；式(3)表示最晚完工时间等于最晚开始加工时间与所需加工时间之和；式(4)表示同一工件上工序加工的先后顺序约束；式(5)表示每个机器任一时刻最多只能加工一道工序；式(6)表示每个工人任一时刻最多只能加工一道工序；式(7)表示所有机器不能提前进行工作，从 0 时刻开始可用；式(8)工件在机器上进行加工时机器必须是空闲的；式(9)机器必须在加工结束后才能停止，中途不能停止；式(10)计算工序最早开始加工时间；式(11)计算工序最晚完工时间。

表 2 为一个 DRCFJSP 问题实例，该实例由三个工件三个机器和两个工人构成。表中给出了每道工序在某个机器上由某个工人加工所需的时间，其中“-”表示对应工人无法操作对应机器对该工序进行加工。例如，$O_{1,1}$ 无法由工人 W_1 在机器 M_1 上进行加工，但 $O_{1,1}$ 可以由工人 W_1 在 M_2 上进行加工且所需加工时间为 1。

表 2　DRCFJSP 问题实例

Tab. 2　Examples of DRCFJSP problems

Job		W_1			W_2		
		M_1	M_2	M_3	M_1	M_2	M_3
J_1	$O_{1,1}$	-	1	-	-	1	6
	$O_{1,2}$	-	2	-	-	2	2
J_2	$O_{2,1}$	5	-	-	-	-	2
	$O_{2,2}$	4	-	-	-	-	3
J_3	$O_{3,1}$	4	6	-	-	6	-
	$O_{3,2}$	2	2	-	-	2	-
	$O_{3,3}$	6	4	-	-	4	1

2　改进布谷鸟算法求解 DRCFJSP

2.1　布谷鸟算法

布谷鸟算法作为一种新型启发式搜索算法，主要基于两个更新策略：一是通过 Lévy 飞行机制寻找寄生巢穴，二是通过偏好随机游走策略代替被发现的鸟巢。在布谷鸟算法中每一个鸟巢代表一个可行解。在理想状态下，布谷鸟的位置更新公式如下：

$$X_i^{t+1}=X_i^t+\alpha\oplus Levy(\beta) \tag{12}$$

其中，X_i^t 表示第 $i(i=1,2,\ldots,n)$ 个鸟巢在第 t 代的位置；$\oplus$ 为点对

点乘法；α 通常取值为 $\alpha=1$。$Levy(\beta)$ 为随机搜索路径，服从 *levy* 分布：

$$Levy(\beta)\sim\mu=t^{-\beta},1<\beta\le3 \tag{13}$$

为方便运算，文献[12]使用式(13)产生 *levy* 随机数：

$$Levy(\beta)=\frac{\phi\times\mu}{|\upsilon|^{\frac{1}{\beta}}} \tag{14}$$

其中，μ 和 υ 均服从标准正态分布，$\beta=1.5$。

$$\phi=\left(\frac{\Gamma(1+\beta)\times\sin(\pi\times\frac{\beta}{2})}{\Gamma((\frac{1+\beta}{2})\times\beta\times2^{\frac{\beta-1}{2}})}\right)^{\frac{1}{\beta}} \tag{15}$$

综合上述公式，布谷鸟的位置更新公式如下：

$$X_i^{t+1}=X_i^t+\alpha_0\times\frac{\phi\times\mu}{|\upsilon|^{\frac{1}{\beta}}}(X_i^t-X_{best}) \tag{16}$$

按发现概率 P_a 丢弃部分解后，采用偏好随机游走重新生成相同数量的新解，公式如下：

$$X_i^{t+1}=X_i^t+\gamma(X_g^t-X_k^t) \tag{17}$$

其中，γ 是服从均匀分布的缩放因子，X_g^t,X_k^t 表示第 t 代的两个随机数。

2.2 算法编码

由于标准布谷鸟算法适用于解决连续性优化问题，无法直接将其用于解决车间调度问题，因此本文引入最小位置规则(smallest position value, SPV)来建立个体与实际调度的映射关系。如图 1 所示，SPV 规则是将原向量进行升序排列得到新向量，新向量中各分量所对应的原向量中的位置即为整数编码序列。本文在算法编码时先随机生成工序加工序列(OS)，然后根据所生成的加工序列，对机器和工人进行分配。整个算法包含了三层序列，第一层是工序加工序列(OS)，第二层是机器分配序列(MA)，第三层是工人分配序列(WA)。该编码由表 1 的问题实例得出，表示 3 个工件由 2 个工人在 3 台机器上加工的调度，其中工件 1 包含 2 道工序、工件 2 包含 2 道工序，工件 3 包含 3 道工序，工件的加工顺序为：$O_{2,1}\rightarrow O_{2,2}\rightarrow O_{3,1}\rightarrow O_{1,1}\rightarrow O_{1,2}\rightarrow O_{3,2}\rightarrow O_{3,3}$。结合整个序列可知，工序 $O_{2,1}$ 由工人 2 在机器 3 上加工，工序 $O_{2,2}$ 由工人 2 在机器 3 上加工，工序 $O_{3,1}$ 由工人 1 在机器 2 上加工，其余工序依此类推。

编码带	1	1	2	2	3	3	3
随机值	0.21	0.37	-0.24	-0.55	0.81	0.66	0.15

SPV ↓ 升序排列

随机值	-0.55	-0.24	0.15	0.21	0.37	0.66	0.81
编码带	2	2	3	1	1	3	3

↓ 分配机器工人

OS	2	2	3	1	1	3	3
	$O_{2,1}$	$O_{2,2}$	$O_{3,1}$	$O_{1,1}$	$O_{1,2}$	$O_{3,2}$	$O_{3,3}$
WA	2	2	1	1	1	2	2
	W_2	W_2	W_1	W_1	W_1	W_2	W_2
MA	3	3	2	2	2	1	3
	M_3	M_3	M_2	M_2	M_2	M_1	M_3

图 1　算法编码过程

Fig. 1　Algorithm coding process

2.3 改进解码算法

相比于单资源约束的 FJSP 问题，DRCFJSP 在加工时不仅受前一道工序的结束时间和机器最早可开始加工时间的约束，还受到工人最早可开始加工时间的约束。因此本文通过对插入式解码方案进行改进，得到一种可以同时对机器和工人进行主动解码的改进解码方案，改进解码方案的核心是有效利用机器和工人加工时产生的空闲时间，将机器和工人提前安排到合适的空闲时间段内进行加工，以达到缩短整个加工时间的目的。记工件 i 的第 j 道工序 O_{ij} 的开始加工时间为 ST_{ij}，结束加工时间为 ET_{ij}，该工序的紧前完工时间为 $ET_{i,j-1}$，工人 s 操作机器 k 对工序 O_{ij} 加工所需时间为 T_{ijks}。在解码时首先需要判断所选机器和工人的加工情况，再根据加工情况判断是否有空闲时间以及空闲时间是否满足插入条件。当机器和工人均有空闲时间时，有以下三种可能出现的情况：

情况一：如图 2 所示，机器和工人的空闲时间不重合，因此无法进行插入操作。

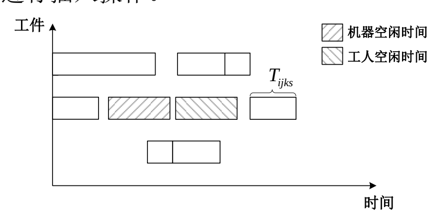


图 2　第一种情况

Fig. 2　The first situation

情况二：如图 3 所示，机器和工人有重合的空闲时间段 T_a，但 T_a 小于工序所需加工时间 T_{ijks}，即 $T_a<T_{ijks}$，因此无法进行插入操作。

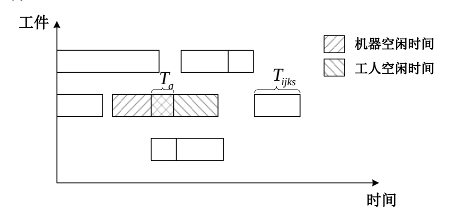


图 3　第二种情况

Fig. 3　The second situation

情况三：如图 4 所示，机器和工人有重合的空闲时间段 T_a，且 T_a 大于等于工序所需加工时间 T_{ijks}，即 $T_a\ge T_{ijks}$，因此可以进行插入操作。

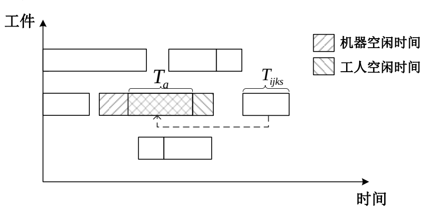


图 4　第三种情况

Fig. 4　The third situation

综合解码时可能出现的所有情况，本文给出了解码算法的伪代码，如算法 1 所示。

算法 1　改进解码方法伪代码

输入：总工序数 TP，机器和工人可加工信息。

输出：工序开始时间 ST_{ij} 和完工时间 ET_{ij}，机器加工时间 $[SM_k,EM_k]$，工人加工时间 $[SW_s,EW_s]$。

For $tp=1$ to TP

　$O_{ij}=OS(tp)$；$flag=0$；　% $flag$ 用于判断是否插入空闲时间

　获取 O_{ij} 可选的加工机器和工人 s，加工所需时间 T_{ijks}；

　机器 k 和工人 s 的加工时间 $[SM_k,EM_k]$、$[SW_s,EW_s]$ 和对应空闲时间 $[ASM_k,AEM_k]$、$[ASW_s,AEW_s]$；

　If $j==1$　%工件第一道工序

　　If $[SM_k,EM_k]==\varnothing$ && $[SW_s,EW_s]==\varnothing$　%工人和机器都还未进行加工

　　　$ST_{ij}=0$；$ET_{ij}=ST_{ij}+T_{ijks}k$　；$flag=1$；

　　End If

　Else

　　If $[ASM_k,AEM_k]\sim=\varnothing$ && $[ASW_s,AEW_s]==\varnothing$　%机器有空闲时间，工人无空闲时间

　　　If $ASM_k>=\max\{EW_s\}$ && $ASM_k>=ET_{i,j-1}$ && $ASM_k+T_{ijks}<=AEM_k$

$ST_{ij}=ASM_k$; $ET_{ij}=ST_{ij}+T_{ijks}$; $flag=1$;
　　End If
　Else If $[ASM_k,AEM_k]==\varnothing$ && $[ASW_s,AEW_s]\sim=\varnothing$ %机器无空闲时间，工人有空闲时间
　　If $ASW_s>=\max\{EM_k\}$ && $ASW_s>=ET_{i,j-1}$ && $ASW_s+T_{ijks}<=AEW_s$
　　　$ST_{ij}=ASW_s$; $ET_{ij}=ST_{ij}+T_{ijks}$; $flag=1$;
　　End If
　Else If $[ASM_k,AEM_k]\sim=\varnothing$ && $[ASW_s,AEW_s]\sim=\varnothing$ %机器和工人均有空闲时间
　　If $\max(ASM_k,ASW_s)>=ET_{i,j-1}$ && $\max(ASM_k,ASW_s)+T_{ijks}<=\min(AEM_k,AEW_s)$
　　　$ST_{ij}=\max\{ASM_k,ASW_s\}$; $ET_{ij}=ST_{ij}+T_{ijks}$; $flag=1$;
　　End If
　Else %机器和工人均无空闲时间
　　$ST_{ij}=\max\{ET_{i,j-1},EM_k,EW_s\}$; $ET_{ij}=ST_{ij}+T_{ijks}$; $flag=1$;
　End If
End If
If $flag==0$
　$ST_{ij}=\max\{ET_{i,j-1},EM_k,EW_s\}$; $ET_{ij}=ST_{ij}+T_{ijks}$; $flag=1$;
End If
更新工序 O_{ij} 的最早完工时间 ET_{ij} 以及机器 k 和工人 s 的加工时间和空闲时间；
End For

2.4　算法更新

2.4.1 改进 Lévy 飞行

在标准布谷鸟搜索算法中，Lévy 飞行的步长越长搜索精度越低，适用于全局探索，步长越短搜索精度越高，局部搜索能力越强。由于标准 Lévy 飞行的步长因子是一个固定值，缺乏自适应性，可能导致算法在搜索时搜索速度慢且容易陷入局部最优。针对这种情况，本文采用两种方法对步长因子进行改进，通过将布谷鸟种群随机划分为三个子群，三个子群分别采用固定步长因子的 Lévy 飞行方式与两种改进自适应步长因子的 Lévy 飞行方式进行独立的更新操作。根据不同搜索阶段种群的搜索情况，自动调整步长因子的大小，以达到平衡全局寻优速率和搜索精度的关系的目的。其中，第一种改进方法如式(18)所示，根据当前鸟巢与当前最优鸟巢的位置关系，对步长因子 α 进行自适应调整，使得 Lévy 飞行在当前最优鸟巢附近进行较为细致的搜索，有利于锁定全局最优解。

$$\alpha=\alpha_0\times(X_i^t-X_{best}) \tag{18}$$

其中，α_0 通常取值为 0.01，X_i^t 表示第 t 代的第 i 个鸟巢位置，X_{best} 表示当前最优鸟巢位置。

第二种改进方法如式(19)所示，采用了文献[18]提出的改进方式，主要通过算法的迭代次数对步长因子 α 进行控制。在算法初期，步长因子 α 的值较大，搜索范围也大，降低了算法陷入局部最优的可能，到了算法后期，步长自适应减小，搜索精度得到了提高，更利于找到最优解：

$$\alpha=(\alpha_{\max}+\alpha_{\gamma})\times\cos\left(\frac{t_i}{t_{\max}}\right) \tag{19}$$

其中，t_i 表示当前迭代次数，$t_{\max}$ 表示迭代总次数，取 $\alpha_{\max}=0.9$ ，α_{γ} 为 $[-0.05,0.05]$ 上随机步长因子。

2.4.2 信息交流

由于三个子群是分别独立的进行寻优操作，为了提高寻优效率，每进化 k 代后，各个种群之间通过一定的策略进行种群间个体的信息交流，以交换种群信息,保持种群的多样性。信息交流的原则是：将子群中的较差个体与最优个体进行信息交流,从而引导较差个体向全局最优个体位置进行搜索。本文引入差分进化算法的 DE/best/1 变异策略对子群进行差分，其计算公式如下：

$$V_{i,g}=X_{best,g}+F(X_{r1,g}-X_{r2,g}) \tag{20}$$

其中，$X_{best,g}$ 表示当前全局最优个体，$X_{r1,g}$ ，$X_{r2,g}$ 表示子群中较差的两个个体。

改进后的布谷鸟算法如算法 2 所示。

算法 2　改进布谷鸟更新伪代码

输入：迭代次数 $iter$ ，子群 $inest$ 。
输出：当前最优鸟巢信息 $X_{best,g}$ 。

For $i=1$ to $iter$
　For $inest=1$ to 3
　　每个子群采用不同步长因子的 Lévy 飞行进行更新，步长因子更新式(12)(18)(19)；
　　对鸟巢进行评价并按照概率 P_a 舍弃较差的鸟巢，采用式(17)生成相同数量的新鸟巢；
　End For
　对这一代鸟巢进行评价，更新当前最优鸟巢信息 $X_{best,g}$ ；
　If $i/k==0$
　　按照式(20)对子群进行差分；
　End If
End For

2.5　算法流程

Step1 初始化参数，布谷鸟种群数目 n ，最大迭代次数 $iter$ ，发现概率 P_a 。

Step2 初始化种群，根据映射规则，将鸟巢信息转换为包含调度信息的可行序列，采用改进解码算法对其进行解码得到完工时间，保留当前最优鸟巢，并随机将种群分为 3 个子群。

Step3 分别采用标准 Lévy 飞行公式和两种改进 Lévy 飞行公式更新子群鸟巢位置，解码得到完工时间，更新当前最优鸟巢。按照式(17)进行偏好随机游走，淘汰部分差的解并生成相同数量的新解。

Step4 每迭代 k 次，引入差分算子来实现三个子群之间的信息交流，用改进解码算法对其进行解码得到完工时间，更新当前最优鸟巢。

Step5 当达到最大迭代次数则输出全局最优解，否则转到 Step3 继续进行迭代。

3　实例仿真与分析

3.1　实验环境和测试实例

本文采用 Matlab R2018a 编程，运行环境为 Intel(R) Xeon(R) CPU E5- @3.30GHz，RAM 8GB。考虑到目前针对 DCRFJSP 的研究成果较少，且没有标准算例可供参考比较，为了更好的通过比较实验结果来验证算法的性能，本文选择 Brandimarte[19]标准测试集中的 MK1-MK10 算例对算法进行测试，同时根据文献[11]中的工人分配方式进行仿真实验，具体工人机器分配情况如表 3 所示。

3.2　结果和分析

由于算法的参数对算法的性能以及运行时间有很大影响，为了保证算法以较优的状态运行，本文引入文献[20]所采用的正交实验法对参数进行设置。图 5 为不同参数情况下，本文算法求解 MK01 算例，运行 10 次的平均值变化趋势图。图 5(a)为最大迭代次数 $iter$ 的影响曲线图，从图中可以看出，当迭代次数设置较小时，可能导致算法在尚未收敛的情况下被迫停止，从而影响算法的寻优结果，随着迭代次数的增加，获得的解的质量也逐渐提高，但是当迭代次数到了某一值后，继续增加迭代次数并不能明显提高取得最优解的次数，还会增加算法的复杂度，因此在保证运行效率的基础上，设置最大迭代次数 $iter=200$ 。图 5(b)可以看出，当种群规模在 50 左

右，能获得比较优秀的解，因此设置种群数量 $n=50$ 。图 5(c) 为发现概率 pa 的影响曲线图，当发现概率在 0.25 左右时，得到的解最优，随着发现概率逐渐增大，得到的解的质量也逐渐降低，因此最终设置发现概率 $pa=0.25$ 。

表 3　工人和机器分配

Tab. 3　Worker and Machine Allocation

Instance	w	Allocation
MK1-2	4	$M(W_1)=\{M_1,M_3,M_5\}$, $M(W_2)=\{M_2,M_4,M_5\}$, $M(W_3)=\{M_1,M_4,M_6\}$, $M(W_4)=\{M_2,M_3,M_4\}$
MK3-4	6	$M(W_1)=\{M_1,M_5\}$, $M(W_2)=\{M_2,M_4\}$, $M(W_3)=\{M_1,M_4,M_6\}$, $M(W_4)=\{M_2,M_3,M_6,M_7\}$, $M(W_5)=\{M_6,M_7,M_8\}$, $M(W_6)=\{M_5,M_8\}$
MK5	3	$M(W_1)=\{M_1,M_3,M_4\}$, $M(W_2)=\{M_2,M_4\}$, $M(W_3)=\{M_1,M_2,M_3\}$
MK6,10	8	$M(W_1)=\{M_1,M_8,M_{10}\}$, $M(W_2)=\{M_2,M_7,M_{11}\}$, $M(W_3)=\{M_3,M_4,M_6,M_{11}\}$, $M(W_4)=\{M_2,M_9,M_{12},M_{13}\}$, $M(W_5)=\{M_6,M_7,M_8,M_{15}\}$, $M(W_6)=\{M_5,M_8,M_{10}\}$, $M(W_7)=\{M_4,M_9,M_{14},M_{15}\}$, $M(W_8)=\{M_1,M_3,M_{10},M_{14}\}$
MK7,11	4	$M(W_1)=\{M_1,M_3,M_5\}$, $M(W_2)=\{M_2,M_4\}$, $M(W_3)=\{M_3,M_4\}$, $M(W_4)=\{M_1,M_2,M_5\}$
MK8,9,12,13	6	$M(W_1)=\{M_1,M_3,M_5\}$, $M(W_2)=\{M_2,M_4,M_9\}$, $M(W_3)=\{M_3,M_4,M_8,M_{10}\}$, $M(W_4)=\{M_1,M_7,M_9\}$, $M(W_5)=\{M_5,M_6,M_7\}$, $M(W_6)=\{M_2,M_4,M_8,M_{10}\}$

图 5　参数影响趋势图

Fig. 5　Parameter influence trend graph

为测试本文算法的有效性，将其与不同的算法进行了比较，每个算例连续充分实验 20 次，记录多次实验的结果并进行分析，仿真实验结果见表 4。其中，*VNS* 、*KGFOA* 和 *MBSA* 分别为文献[10]、文献[11]和文献[21]中提出的算法求解 DRCFJSP 所得的最优值； *ICS* 为普通解码方式的改进布谷鸟算法求解所得的最优值； *CSND* 为具有改进解码方式的标准布谷鸟算法求解所得的最优值；*ICSND* 为本文采用的具有改进解码方式的改进布谷鸟算法求解所得的最优值。

从表 4 可以看出，在仅对布谷鸟算法进行改进的情况下，当问题规模较小时，容易得到比较优秀的值；但是当问题规模逐渐增大，机器和工人空闲时间也随之增多，普通的解码方式无法有效利用这些空闲时间，因此仅靠改进布谷鸟算法已经无法得到更好的解。对于 *CSND* ，由于布谷鸟算法本身具有一定的优越性，能较好地兼顾全局和局部搜索，再加上改进解码方式有效的缩短了完工时间，求解的效率也得到了一定的提升。对于本文所使用的 *ICSND* ，不仅对布谷鸟算法进行了改进，增强了算法的搜索能力，还对解码方式进行了调整，有效利用了工人和机器的空闲时间，极大的缩短了完工时间，将 *ICSND* 与其他算法进行对比，可以发现在大部分算例的求解上， *ICSND* 都能得到更优的解。

表 4　不同算法求解 Brandimarte 算例的结果对比

Tab. 4　Comparison of the results of different algorithms for solving the Brandimarte example

Instance	*VNS*	*KGFOA*	*MBSA*	*ICS*	*CSND*	*ICSND*
MK1	63	61	47	49	46	**44**
MK2	51	52	39	40	40	**38**
MK3	204	204	204	213	204	**204**
MK4	69	67	79	80	75	**67**
MK5	337	287	239	244	237	**235**
MK6	89	86	75	82	76	**73**
MK7	184	184	185	198	185	**181**
MK8	**536**	551	561	612	572	557
MK9	437	**407**	**407**	457	436	430
MK10	328	315	311	332	292	**290**

图 6 和图 7 为采用本文 *ICSND* 算法求解算例 MK1 和 MK2 所得的甘特图，其中，横坐标为加工时长，纵坐标为机器，方框里的第一个数字表示工件，第二个数字表示对应的工人。

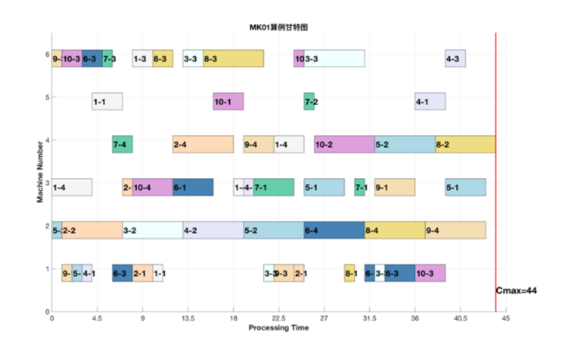


图 6　MK1 甘特图

Fig. 6　MK1 gantt chart

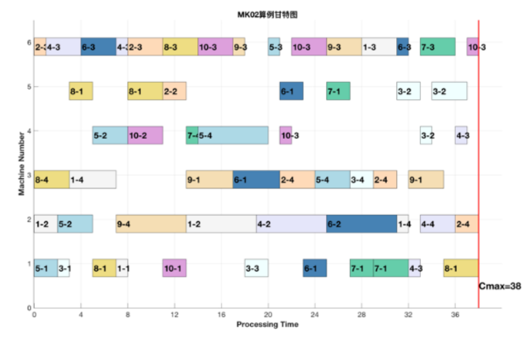


图 7　MK2 甘特图

Fig. 7　MK2 gantt chart

为了更好的衡量本文所使用的算法性能，引入文献[10]所使用的相对百分比偏差最优值 *MRPD* 和相对百分比偏差平均值 *ARPD* 这两个指标来评估算法能力，计算公式为

$$MRPD=\min\left(\frac{C_{\max}-C_{\max}^{low}}{C_{\max}^{low}}\times 100\right) \tag{21}$$

$$ARPD=\frac{1}{S}\sum_{i=1}^{S}\left(\frac{C_{\max}-C_{\max}^{low}}{C_{\max}^{low}}\times 100\right) \tag{22}$$

其中，$C_{\max}$ 为各个算法求出的最优解，$C_{\max}^{low}$ 为本文求出的最优解。

从表 5 可以看出，采用了改进解码方式的算法 *MRPD* 和 *ARPD* 都优于普通解码方式的算法，而 *ICSND* 无论从解的质量还是最优解的获取都优于 *CSND* ，改进后的布谷鸟算法性能整体优于标准布谷鸟算法，具有更强的搜索能力。表 6 通过将 *ICSND* 与 *VNS* 、*KGFOA* 和 *MBSA* 的实验结果进行对比可以看出，*ICSND* 的最小百分比偏差都为 0，并且平均百分比偏差也更小，得到的解更加稳定，质量也比较好。而其他三种算法，在大部分情况下的求解质量都不如 *ICSND* 算法，平均百分比偏差的值也更大。因此相对于其他三种算法， *ICSND* 算法具

有更好的求解能力，在求解 DRCFJSP 问题时的稳定性和解的质量都更优。

表 5 不同布谷鸟算法之间的对比分析

Tab. 5 Comparative analysis between different cuckoo algorithms

Instance	C_{max}^{low}	*ICSND*		*ICSND*		*ICS*	
		MRPD	*MRPD*	*MRPD*	*ARPD*	*MRPD*	*ARPD*
MK1	44	0	3.11	0	6.44	8.89	14.67
MK2	38	0	2.37	2.63	4.2	5.26	10.26
MK3	204	0	2.45	0	3.34	5.39	10.69
MK4	67	0	4.10	10.45	18.20	20.89	24.93
MK5	235	0	8.89	1.70	2.46	4.68	6.09
MK6	73	0	2.73	2.73	4.52	13.69	16.98
MK7	181	0	2.70	2.21	3.20	6.07	9.66
MK8	557	0	2.52	2.49	4.12	8.71	12.72
MK9	430	0	1.03	0.69	2.14	10.80	12.48
MK10	290	0	0.89	0.34	1.44	9.31	11.31

表 6 布谷鸟算法和其他三种算法的对比分析

Tab. 6 Comparative analysis of the cuckoo algorithm and the other three algorithms

Instance	$m\times n\times w$	C_{max}^{low}	*ICSND*		*VNS*		*KGFOA*		*MBSA*	
			MRPD	*ARPD*	*MRPD*	*ARPD*	*MRPD*	*ARPD*	*MRPD*	*ARPD*
MK1	10×6×4	44	0	3.11	35.55	58.76	40.00	46.67	0	5.11
MK2	10×6×4	38	0	2.37	36.84	68.38	34.21	47.36	0	2.89
MK3	15×8×6	204	0	2.45	0	49.18	0	41.00	0	4.49
MK4	15×8×6	67	0	4.10	0	31.76	-11.94	26.28	5.94	15.52
MK5	15×4×3	235	0	8.89	22.12	38.70	43.40	33.37	1.27	2.04
MK6	10×15×8	73	0	2.73	17.80	70.44	21.91	61.27	1.36	3.56
MK7	20×5×4	181	0	2.70	1.65	10.19	1.65	2.73	2.79	5.68
MK8	20×10×6	557	0	2.52	-1.95	20.79	-4.62	16.70	1.54	4.77
MK9	20×10×6	430	0	1.03	-6.43	26.98	0.45	19.91	3.41	9.84
MK10	20×15×8	290	0	0.89	8.62	56.55	13.10	48.91	3.80	8.47

4 结束语

本文在布谷鸟算法核心框架不变的基础下对其进行改进，通过将布谷鸟种群划分为 3 个子群，并分别采用不同 Lévy 飞行方式进行寻优，同时引入差分算子进行子群间的信息交流，大大增加了算法的搜索能力。在解码时设计了一种改进解码方式，在避免机器和工人的加工时间冲突的同时，主动寻找可进行插入的空闲时间，大大的缩短了整个加工的完工时间。最后通过基准测试算例进行实验验证和比较，证明了改进布谷鸟算法和改进解码方式的有效性和稳定性。在后续研究中，考虑将生产过程中常见的动态因素考虑进去，比如机器故障、机器维护、紧急插单以及紧急撤单等。

参考文献：

[1] 陈可嘉，段瑞明，刘碧玉，等. 模糊置换流水车间调度的多目标模型 [J]. 运筹与管理, 2021, 30 (08): 28-36. (Chen Kejia, Duan Ruiming, Liu Biyu, *et al.* A multi-objective model for fuzzy replacement flow shop scheduling [J]. Operations Research and Management, 2021, 30 (08): 28-36.)

[2] Lin J. Backtracking search based hyper-heuristic for the flexible job-shop scheduling problem with fuzzy processing time [J]. Engineering Applications of Artificial Intelligence, 2019, 77: 186-196.

[3] 轩华，王晶，李冰，等. 阻塞混合流水车间调度优化研究 [J]. 控制工程, 2020, 27 (08): 1346-1350. (Xuan Hua, Wang Jing, Li Bing, *et al.* Research on Optimal Scheduling of Blocked Mixed Flow Shop [J]. Control Engineering, 2020, 27 (08): 1346-1350.)

[4] Shao Z, Shao W, Pi D. Effective constructive heuristic and iterated greedy algorithm for distributed mixed blocking permutation flow-shop scheduling problem [J]. Knowledge-Based Systems, 2021, 221 (5): 106959.

[5] 轩华，王晶，张慧贤，等. 混合遗传算法求解含机器可利用约束的 HFSP [J]. 计算机应用与软件, 2021, 38 (06): 176-181. (Xuan Hua, Wang Jing, Zhang Huixian, *et al.* Hybrid Genetic Algorithm for HFSP with Machine Availability Constraints [J]. Computer Applications and Software, 2021, 38 (06): 176-181.)

[6] 张洪亮，丁仁曼，徐公杰. 考虑区间工时的多目标柔性作业车间节能调度 [J/OL]. 系统仿真学报: 1-13 (2021-08-17) [2021-09-10]. https://doi. org/10. 16182/j. issn1004731x. joss. 21-0395. (Zhang Hongliang, Ding Renman, Xu Gongjie. Multi-objective Flexible Job Shop Energy Saving Scheduling Considering Interval Working Hours [J/OL]. Journal of System Simulation: 1-13 (2021-08-17) [2021-09-10]. https://doi. org/10. 16182/j. issn1004731x. joss. 21-0395.)

[7] Wang H, Sheng B, Lu Q, *et al.* A novel multi-objective optimization algorithm for the integrated scheduling of flexible job shops considering preventive maintenance activities and transportation processes [J]. Soft Computing, 2021, 25 (4): 1-27.

[8] Li J, Yuan H. A Hybrid Genetic Algorithm for Dual-Resource Constrained Job Shop Scheduling Problem [J]. Computers & Industrial Engineering, 2016, 102: 113-131.

[9] Jing Z, Wang W, Xu X. A hybrid discrete particle swarm optimization for dual-resource constrained job shop scheduling with resource flexibility [J]. Journal of Intelligent Manufacturing, 2017, 28 (8): 1961-1972.

[10] Lei D, Guo X. Variable neighbourhood search for dual-resource constrained flexible job shop scheduling [J]. International Journal of Production Research, 2014, 52 (9): 2519-2529.

[11] Zheng X L, Wang L. A knowledge-guided fruit fly optimization algorithm for dual resource constrained flexible job-shop scheduling problem [J]. International Journal of Production Research, 2016, 54 (18): 5554-5566.

[12] Yang X S, DEB S. Cuckoo search via Lévy flight [C]// Proceedings of World Congress on Nature & Biologically Inspired Computing. India Washington: IEEE Publications, 2009: 210-214.

[13] Ouaarab A, Ahiod B, Yang X S. Discrete Cuckoo Search Applied to Job Shop Scheduling Problem [M]. Recent Advances in Swarm Intelligence and Evolutionary Computation. Springer International Publishing, 2015: 121-137.

[14] Alaa S, Alobaidi A. Two Improved Cuckoo Search Algorithms for Solving The Flexible Job-Shop Scheduling Problem [J]. International Journal on Perceptive and Cognitive Computing, 2016, 2 (2): 25-31.

[15] Majumdera A, Lahaa D, Suganthan P N. A hybrid cuckoo search algorithm in parallel batch processing machines with unequal job ready times [J]. Computers & Industrial Engineering, 2018, 124: 65-76.

[16] 唐红涛，刘家毅. 改进的布谷鸟算法求解考虑运输时间的分布式柔性流水车间调度问题 [J]. 运筹与管理, 2021, 30 (11): 76-83. (Tang Hongtao, Liu Jiayi. Improved Cuckoo Algorithm for Distributed Flexible Flow Shop Scheduling Problem Considering Transportation Time [J]. Operations Research and Management, 2021, 30 (11): 76-83.)

[17] 罗函明，罗天洪，吴晓东，等. 求解混合流水车间调度问题的离散布谷鸟算法 [J]. 计算机工程与应用, 2020, 56 (22): 264-271. (Luo Hanming, Luo Tianhong, Wu Xiaodong, *et al.* Discrete Cuckoo Algorithm for Solving Hybrid Flow Shop Scheduling Problem [J]. Computer Engineering and Applications, 2020, 56 (22): 264-271.)

[18] 施文章，韩伟，戴睿闻. 模拟退火下布谷鸟算法求解车间作业调度问题 [J]. 计算机工程与应用, 2017, 53 (17): 249-253, 259. (Shi Wenzhang, Han Wei, Dai Ruiwen. Cuckoo algorithm under simulated

annealing to solve job shop scheduling problems [J]. Computer Engineering and Applications, 2017, 53 (17): 249-253, 259.)

[19] Brandimarte P. Routing and scheduling in a flexible job shop by tabu search [J]. Annals of Operations research, 1993, 41 (3): 157-183.

[20] 王文艳, 徐震浩, 顾幸生. 离散水波优化算法求解带批处理的混合流水车间批量流调度问题 [J]. 华东理工大学学报: 自然科学版, 2021, 47 (05): 598-608. (Wang Wenyan, Xu Zhenhao, Gu Xingsheng. Discrete water wave optimization algorithm for batch flow scheduling problem in mixed flow workshop with batch processing [J]. Journal of East China University of Science and Technology: Natural Science, 2021, 47 (05): 598-608.)

[21] Gnanavelbabu A, Caldeira R H, Vaidyanathan T. A simulation-based modified backtracking search algorithm for multi-objective stochastic flexible job shop scheduling problem with worker flexibility [J]. Applied Soft Computing, 2021, 2021 (113): 107960.